Automates avancés – Partiel (durée : 1h45)

Tous les documents sont autorisés.

Le barème est donné à titre indicatif et peut être modifié.

Quelques rappels de définitions (déjà tous vues en TD) : Une *grammaire hors-contexte* est un quadruplet $G = (V, T, P, S)$, où V et T sont deux ensembles finis disjoints (appelés respectivement variables (ou non-terminaux) et terminaux), P est un ensemble de productions de la forme $A \to \alpha$, où $A \in V$ et $\alpha \in (V \cup T)^*$ et S est le symbole de départ. Soit $\Sigma = V \cup T$.

Nous définissons deux relations $\Rightarrow$ et $\Rightarrow_G^*$ entre suites de Σ^* comme suit : Si $A \to \beta$ est une production de P et $\alpha, \gamma \in \Sigma^*$, $\alpha A \gamma \Rightarrow \alpha\beta\gamma$. La suite $\alpha A \gamma$ est un *prédécesseur immédiat* de $\alpha\beta\gamma$ et $\alpha\beta\gamma$ est un *successeur immédiat* de $\alpha A \gamma$. La relation $\Rightarrow^*$ est alors la fermeture réflexive et transitive de $\Rightarrow$. Si $\alpha \Rightarrow^* \beta$, alors α est un prédécesseur de β et β est un successeur de α. Étant donné $L \subseteq \Sigma^*$, nous définissons $pre(L) = \{\alpha \in \Sigma^* \mid \exists \beta \in L \text{ avec } \alpha \Rightarrow \beta\}$ et $post(L) = \{\beta \in \Sigma^* \mid \exists \alpha \in L \text{ avec } \alpha \Rightarrow \beta\}$. $pre^i(L)$ est inductivement défini par $pre^0(L) = L$ et $pre^{i+1}(L) = pre(pre^i(L))$. Enfin, nous définissons $pre^*(L) = \bigcup_{i \geq 0} pre^i(L)$ ce qui est le même que $\{\alpha \in \Sigma^* \mid \exists \beta \in L \text{ avec } \alpha \Rightarrow^* \beta\}$. De même, $post^*(L) = \{\beta \in \Sigma^* \mid \exists \alpha \in L \text{ avec } \alpha \Rightarrow^* \beta\}$. Le langage généré par G est donné par $L(G) = post^*(\{S\}) \cap T^*$.

Un algorithme simple pour calculer $pre^*(L)$ d'un langage régulier donné par un automate M est donné ci-dessous. Il est basé sur la saturation de la relation de transition d'un automate non-déterministe $M = (Q, \Sigma, \delta, q_0, F)$, où $\delta \subseteq Q \times \Sigma \times Q$. Nous définissons la relation de transition étendue $\widehat{\delta} : (Q \times \Sigma^*) \to 2^Q$ par : $\widehat{\delta}(q, \epsilon) = \{q\}$, $\widehat{\delta}(q, a) = \{q' \mid (q, a, q') \in \delta\}$ et $\widehat{\delta}(q, wa) = \{p \mid p \in \widehat{\delta}(r, a) \text{ pour un } r \in \widehat{\delta}(q, w)\}$

Entrée : $G = (V, T, P, S), M = (Q, \Sigma, \delta, q_0, F)$, **Sortie** : δ_{pre^*}
$rel \leftarrow \delta$
Répéter
  pour $q, q' \in Q, A \to \beta \in P$ **faire**
    si $q' \in \widehat{rel}(q, \beta)$ **alors** $rel \leftarrow rel \cup \{(q, A, q')\}$
jusqu'à ce que rel ne change plus
retourner rel

Exercice 1 [Grammaires hors-contexte, 7 points]

Soit $G = (V, T, P, S)$ une grammaire hors-contexte **sans** symboles inutiles. Soit $\Sigma = V \cup T$. Le langage $L(G)$ généré par G est infini (c.-à-d. elle contient un nombre infini de mots) si et seulement si il existe un $X \in V$, tel que $X \Rightarrow^* \alpha X \beta$ avec $\alpha, \beta \in \Sigma^*$ et $\alpha\beta \neq \epsilon$ (c'est-à-dire $\alpha \neq \epsilon$ ou $\beta \neq \epsilon$).

- Donnez un algorithme basé sur pre^* pour savoir si une grammaire génère un langage infini.

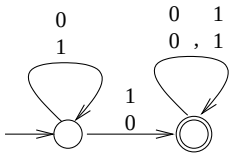
- Considérez la grammaire suivante :

$$S \to ABa \mid CA \mid$$
$$B \to CC \mid B$$
$$A \to a$$
$$C \to a \mid b$$
$$B \to d$$

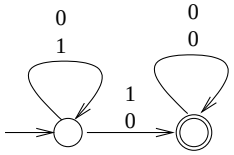
  - Appliquez votre algorithme à cette grammaire pour savoir si elle génère un langage infini.
  - Est-ce que aaa est généré par la grammaire ? Calculez pour cela $pre^*(aaa)$.

Exercice 2 [Automates pour l'arithmétique de Presburger, 7 points] Comme habituellement, on suppose qu'on lit les entiers non-négatifs en binaire avec les bits les plus faibles d'abord.

- Donnez un automate correspondant aux solutions de l'équation $3x - 5y = 1$. Indication : L'automate a 7 états (sans état puits).
- Considérez l'automate suivant sur l'alphabet $\{\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 \\ 1 \end{pmatrix}\}$.



  Un mot accepté par l'automate est par exemple $\begin{pmatrix} 0 \\ 1 \end{pmatrix}\begin{pmatrix} 0 \\ 1 \end{pmatrix}\begin{pmatrix} 1 \\ 0 \end{pmatrix}$ ce qui correspond à $(4, 3)$. Donnez une formule de la logique de Presburger dont les solutions correspondent exactement aux mots acceptés par l'automate.
- Considérez maintenant l'automate suivant :



  Donnez une formule logique dont les solutions correspondent exactement aux mots acceptés par l'automate. Est-ce qu'on peut donner une formule de la logique de Presburger dont les solutions correspondent exactement aux mots acceptés par l'automate ?

Exercice 3 [Image de Parikh, 7 points]

Soit une grammaire hors-contexte G donnée par les productions $S \to BA \mid CCA$, $B \to BAAA \mid a, C \to bCcC \mid cCbC \mid \epsilon, A \to b$

- Donnez une formule de Presburger (formule arithmétique linéaire) qui représente l'image de Parikh de $L(G)$. Construisez cette formule directement à partir de G.
- Donnez un automate fini dont le langage a le même image de Parikh.
- Quel est $L(G)$?
- Montrez que $L(G)$ n'est pas régulier.